Building Startup Systems

Objective

1. Setup a working toolchain for *React Native*
2. Grok the mental model for *React Native*
3. Build and deploy a simple working *React Native* application
4. Understand best practices + future directions

Tools - The Basics

To Build React Native Apps

- ➔ Download Expo XDE at expo.io/tools
  Expo XDE is a GUI tool for packaging and deploying apps
- ➔ Install Watchman
  https://facebook.github.io/watchman/docs/install.html
  Watchman is a tool that takes actions on-file-change, needed by React
- ➔ Ensure you have npm v4 installed
  `npm i npm@4 -g`

Tools - The Device

To Test Using Your Own Phone

➔ Download the Expo App for your phone
Expo App runs on you phone and can demo apps pre-deployment

To Develop Using an Android Emulator

➔ Install genymotion personal
https://www.genymotion.com/fun-zone/

➔ Install the Android SDK
`brew cask install android-sdk`

➔ `Refer to Instructions if you get stuck`
https://facebook.github.io/react-native/releases/0.23/docs/android-setup.html

Tools - The Language

React Native is written in ES6 and cross-compiled using babel.js

- ES6 adds makes JavaScript much easier to write!
- Modules, Classes, Exports and Imports
- Arrow functions

```
let l = x => x + 1;
```

- let, const vs. var
- async + await

Let's Do It

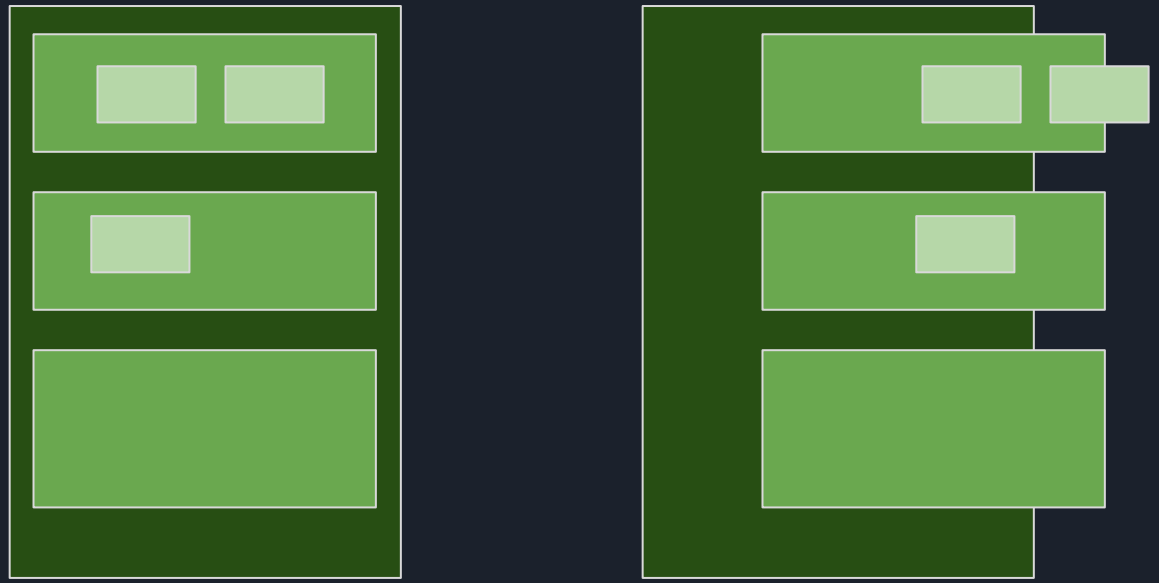
```
# Only need to do this one time ever.
# Installs the create-react-native-app tool

$ npm install -g create-react-native-app
```

```
# Do this to initialize a new react project

$ create-react-native-app my_app
```

JSX for React Native

```
// Components must support a render() function!

class MyComponent extends React.Component {
    render() {
      return (
         // what goes here?
      );
    }
}
```

Think of your UI as composed elements



```
<View>
  <View>
    <Image/>
    <Image/>
  </View>

  <View>
    <Text/>
  </View>

  <TextInput/>
</View>
```

References as you Develop

Comprehensive list of Components and Properties

facebook.github.io/react-native/docs

Excellent Walkthrough of RN Core Concepts, Redux + More

www.reactnativeexpress.com

React-Native Best Practices

- **Components** are used to organize and construct the UI
  Smart Components are aware of the data and logic unique to your application. Containers pass data and callbacks as props to presentational components, and handle updating the data when a user interacts with the app.
  Presentational Components are presentation-only, styled components, that defer to their smarter partners for processing. Their only input is their props
- **Views** are primarily used for styling and layout of children elements
- **Styles** configure views - layout, spacing, colors, etc
  see http://www.reactnativeexpress.com/view
- In React it's a convention to export one component from a file, and to export it is as the **default** export.